

# Package ‘mnormt’

February 14, 2012

**Version** 1.4-5

**Date** 2012-01-06

**Title** The multivariate normal and t distributions

**Author** Fortran code by Alan Genz, R code by Adelchi Azzalini

**Maintainer** Adelchi Azzalini <azzalini@stat.unipd.it>

**Depends** R (>= 2.2.0)

**Description** This package provides functions for computing the density and the distribution function of multivariate normal and multivariate “t” variates, and for generating random vectors sampled from these distributions. Probabilities are computed via a non-Monte Carlo method; different routines are used for the case  $d=1$ ,  $d=2$ ,  $d>2$ , if  $d$  denotes the number of dimensions.

**License** GPL-2

**URL** <http://azzalini.stat.unipd.it/SW/Pkg-mnormt>

**Repository** CRAN

**Date/Publication** 2012-01-07 05:45:19

## R topics documented:

dmnorm	2
dmt	4
pd.solve	6
<b>Index</b>	<b>7</b>

dmnorm

*Multivariate normal distribution***Description**

The probability density function, the distribution function and random number generation for the multivariate normal (Gaussian) probability distribution

**Usage**

```
dmnorm(x, mean = rep(0, d), varcov, log = FALSE)
pmnorm(x, mean = rep(0, length(x)), varcov, ...)
rmnorm(n = 1, mean = rep(0, d), varcov)
sadmvn(lower, upper, mean, varcov, maxpts = 2000 * d, abseps = 1e-06, releps = 0)
```

**Arguments**

x	for dmnorm, this is either a vector of length d or a matrix with d columns, where $d = \text{ncol}(\text{varcov})$ , giving the coordinates of the point(s) where the density must be evaluated; for pmnorm, only a vector of length d is allowed, and d cannot exceed 20
mean	a numeric vector representing the expected value of the distribution; it must be of length d, as defined above. For dmnorm it can be a matrix; in this case its dimensions must match those of x
varcov	a positive definite matrix representing the variance-covariance matrix of the distribution; a vector of length 1 is also allowed (in this case, d=1 is set)
log	a logical value (default value is FALSE); if TRUE, the logarithm of the density is computed
...	parameters passed to sadmvn, among maxpts, abseps, releps
n	the number of random numbers to be generated
lower	a numeric vector of lower integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed
upper	a numeric vector of upper integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed
maxpts	the maximum number of function evaluations (default value: 2000*d)
abseps	absolute error tolerance (default value: 1e-6)
releps	relative error tolerance (default value: 0)

**Details**

The function pmnorm works by making a suitable call to sadmvn if  $d > 2$ , or to `biv.nt.prob` if  $d = 2$ , or to `pnorm` if  $d = 1$ . Function sadmvn is an interface to a Fortran-77 routine with the same name written by Alan Genz, and available from his web page; this makes uses of some auxiliary functions whose authors are documented in the Fortran code. The routine uses an adaptive integration method.

**Value**

dmnorm returns a vector of density values (possibly log-transformed); pmnorm and sadmvn return a single probability with attributes giving details on the achieved accuracy; rmnorm returns a matrix of n rows of random vectors

**Note**

The attributes error and status of the probability returned by pmnorm and sadmvn indicate whether the function had a normal termination, achieving the required accuracy. If this is not the case, re-run the function with an higher value of maxpts

**Author(s)**

Fortran code of SADMVN and most auxiliary functions by Alan Genz, some additional auxiliary functions by people referred to within his program. Porting to R and additional R code by Adelchi Azzalini

**References**

Genz, A. (1992). Numerical Computation of Multivariate Normal Probabilities. *J. Computational and Graphical Statist.*, **1**, 141-149.

Genz, A. (1993). Comparison of methods for the computation of multivariate normal probabilities. *Computing Science and Statistics*, **25**, 400-405.

Genz, A.: Fortran code available at <http://www.math.wsu.edu/math/faculty/genz/software/fort77/mvn.f>

**See Also**

[dnorm](#), [dmt](#), [biv.nt.prob](#)

**Examples**

```
x <- seq(-2,4,length=21)
y <- 2*x+10
z <- x+cos(y)
mu <- c(1,12,2)
Sigma <- matrix(c(1,2,0,2,5,0.5,0,0.5,3), 3, 3)
f <- dmnorm(cbind(x,y,z), mu, Sigma)
p1 <- pmnorm(c(2,11,3), mu, Sigma)
p2 <- pmnorm(c(2,11,3), mu, Sigma, maxpts=10000, abseps=1e-10)
x <- rmnorm(10, mu, Sigma)
p <- sadmvn(lower=c(2,11,3), upper=rep(Inf,3), mu, Sigma) # upper tail
#
p0 <- pmnorm(c(2,11), mu[1:2], Sigma[1:2,1:2])
p1 <- biv.nt.prob(0, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
p2 <- sadmvn(lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
c(p0, p1, p2, p0-p1, p0-p2)
#
p1 <- pnorm(0, 1, 3)
p2 <- pmnorm(0, 1, 3^2)
```

dmt

*Multivariate t distribution***Description**

The probability density function, the distribution function and random number generation for the multivariate t probability distribution

**Usage**

```
dmt(x, mean = rep(0, d), S, df=Inf, log = FALSE)
pmt(x, mean = rep(0, length(x)), S, df=Inf, ...)
rmt(n = 1, mean = rep(0, d), S, df=Inf)
sadmvt(df, lower, upper, mean, S, maxpts = 2000 * d, abseps = 1e-06, releps = 0)
biv.nt.prob(df, lower, upper, mean, S)
```

**Arguments**

x	for dmt, this is either a vector of length d or a matrix with d columns, where $d = \text{ncol}(S)$ , giving the coordinates of the point(s) where the density must be evaluated; for pmt, only a vector of length d is allowed, and d cannot exceed 20
mean	a numeric vector representing the location parameter of the distribution (equal to the expected value when $df > 1$ ); it must be of length d, as defined above. For dmt it can be a matrix; in this case its dimensions must match those of x
S	a positive definite matrix representing the scale matrix of the distribution, such that $S * df / (df - 2)$ is the variance-covariance matrix when $df > 2$ ; a vector of length 1 is also allowed (in this case, $d = 1$ is set)
df	degrees of freedom; it must be a positive integer for pmt, sadmvt and biv.nt.prob, otherwise a positive number. If $df = \text{Inf}$ (default value), the corresponding <i>*mnorm</i> function is called, unless $d = 2$ ; in this case biv.nt.prob is used. If biv.nt.prob is called with $df = \text{Inf}$ , it returns the probability of a rectangle assigned by a bivariate normal distribution
log	a logical value; if TRUE, the logarithm of the density is computed
...	parameters passed to sadmvt, among maxpts, absrel, releps
n	the number of random numbers to be generated
lower	a numeric vector of lower integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed
upper	a numeric vector of upper integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed
maxpts	the maximum number of function evaluations (default value: $2000 * d$ )
abseps	absolute error tolerance (default value: $1e-6$ )
releps	relative error tolerance (default value: 0)

## Details

The functions `sadmvt` and `biv.nt.prob` are interfaces to Fortran-77 routines by Alan Genz, and available from his web page; they makes uses of some auxiliary functions whose authors are documented in the Fortran code. The routine `sadmvt` uses an adaptive integration method. The routine `biv.nt.prob` is specific for the bivariate case; if `df<1` or `df=Inf`, it computes the bivariate normal distribution function using a non-iterative method described in a reference given below. If `pmt` is called with `d>2`, this is converted into a suitable call to `sadmvt`; if `d=2`, a call to `biv.nt.prob` is used; if `d=1`, then `pt` is used.

## Value

`dmt` returns a vector of density values (possibly log-transformed); `pmt` and `sadmvt` return a single probability with attributes giving details on the achieved accuracy; `rmt` returns a matrix of `n` rows of random vectors

## Note

The attributes `error` and `status` of the probability returned by `pmt` and `sadmvt` indicate whether the function had a normal termination, achieving the required accuracy. If this is not the case, re-run the function with an higher value of `maxpts`

## Author(s)

Fortran code of `SADMVT` and most auxiliary functions by Alan Genz, some additional auxiliary functions by people referred to within his program. Porting to R and additional R code by Adelchi Azzalini

## References

Genz, A.: Fortran code in files `mvt.f` and `mvtdstpack.f` available at <http://www.math.wsu.edu/math/faculty/genz/software/>

Dunnnett, C.W. and Sobel, M. (1954). A bivariate generalization of Student's *t*-distribution with tables for certain special cases. *Biometrika* 41, 153–169.

## See Also

[dt](#), [dmnorm](#)

## Examples

```
x <- seq(-2,4,length=21)
y <- 2*x+10
z <- x+cos(y)
mu <- c(1,12,2)
Sigma <- matrix(c(1,2,0,2,5,0.5,0,0.5,3), 3, 3)
df <- 4
f <- dmt(cbind(x,y,z), mu, Sigma,df)
p1 <- pmt(c(2,11,3), mu, Sigma, df)
p2 <- pmt(c(2,11,3), mu, Sigma, df, maxpts=10000, abseps=1e-8)
x <- rmt(10, mu, Sigma, df)
```

```

p <- sadmvt(df, lower=c(2,11,3), upper=rep(Inf,3), mu, Sigma) # upper tail
#
p0 <- pmt(c(2,11), mu[1:2], Sigma[1:2,1:2], df=5)
p1 <- biv.nt.prob(5, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
p2 <- sadmvt(5, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
c(p0, p1, p2, p0-p1, p0-p2)

```

---

pd.solve

*Inverse of a positive definite matrix*

---

### Description

The inverse of a symmetric positive definite matrix and its log-determinant

### Usage

```
pd.solve(x, silent = FALSE, log.det=FALSE)
```

### Arguments

x	a symmetric positive definite matrix
silent	a logical value which indicates the action to take in case of an error. If <code>silent==TRUE</code> and an error occurs, the function silently returns a NULL value; if <code>silent==FALSE</code> (default) an error generates a stop with an error message
log.det	a logical value to indicate whether the log-determinant of x is required (default is FALSE)

### Details

The function checks that x is a symmetric positive definite matrix. If an error is detected, an action is taken which depends on the value of the argument `silent`.

### Value

the inverse matrix of x; if `log.det=TRUE`, this inverse has an attribute which contains the logarithm of the determinant of x

### Author(s)

Adelchi Azzalini

### Examples

```

x <- toeplitz(rev(1:4))
x.inv <- pd.solve(x)
print(x.inv %**% x)
x.inv <- pd.solve(x, log.det=TRUE)
logDet <- attr(x.inv, "log.det")
print(abs(logDet - determinant(x, logarithm=TRUE)$modulus))

```

# Index

\*Topic **algebra**

pd.solve, 6

\*Topic **array**

pd.solve, 6

\*Topic **distribution**

dmnorm, 2

dmt, 4

\*Topic **multivariate**

dmnorm, 2

dmt, 4

biv.nt.prob, 3

biv.nt.prob (dmt), 4

dmnorm, 2, 5

dmt, 3, 4

dnorm, 3

dt, 5

pd.solve, 6

pmnorm (dmnorm), 2

pmt (dmt), 4

rmnorm (dmnorm), 2

rmt (dmt), 4

sadmvn (dmnorm), 2

sadmvt (dmt), 4